


RESEARCH ARTICLE

# Picking out the Impurities: Attention-based Push-Grasping in Dense Clutter

Ning Lu<sup>1</sup>, Yinghao Cai<sup>2</sup>, Tao Lu<sup>2,\*</sup> , Xiaoge Cao<sup>2</sup>, Weiyan Guo<sup>2</sup> and Shuo Wang<sup>2,3</sup>

<sup>1</sup>China Waterborne Transport Research Institute, Beijing, 100088, China, <sup>2</sup>The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China, and <sup>3</sup>Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Shanghai, 200031, China

\*Corresponding author. E-mail: [tao.lu@ia.ac.cn](mailto:tao.lu@ia.ac.cn)

**Received:** 16 July 2021; **Revised:** 15 January 2022; **Accepted:** 9 February 2022; **First published online:** 25 March 2022

**Keywords:** saliency detection, robot learning, push–grasp, target-oriented grasping, deep Q network

## Abstract

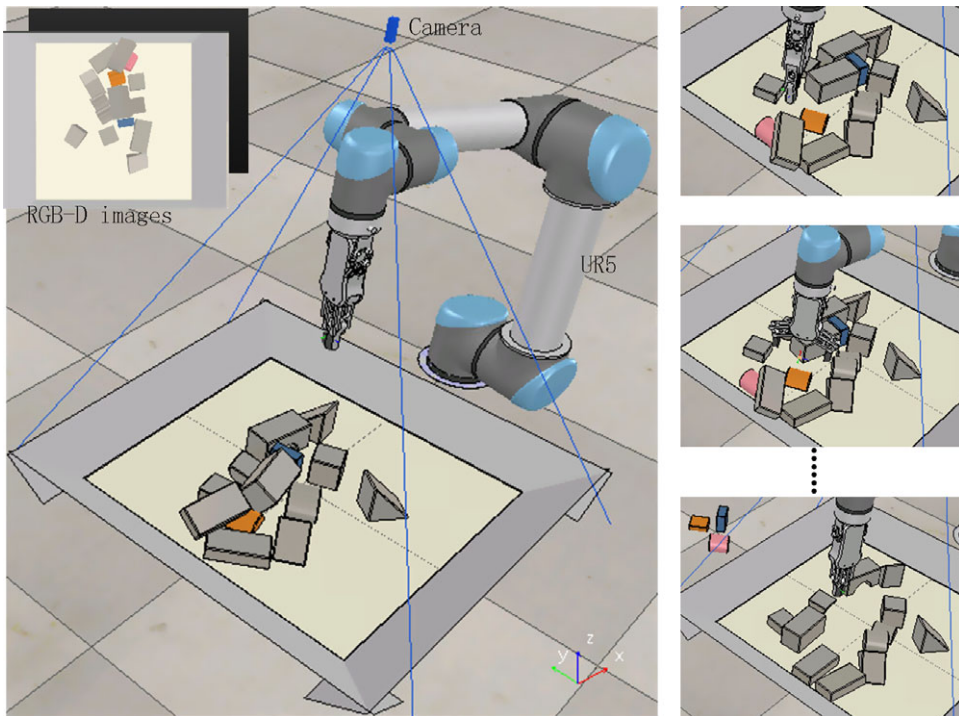
“Picking out the impurities” is a typical scenario in production line which is both time consuming and laborious. In this article, we propose a target-oriented robotic push-grasping system which is able to actively discover and pick the impurities in dense environments with the synergies between pushing and grasping actions. First, we propose an attention module, which includes target saliency detection and density-based occluded-region inference. Without the necessity of expensive labeling of semantic segmentation, our attention module can quickly locate the targets in the view or predict the candidate regions where the targets are most likely to be occluded. Second, we propose a push–grasp synergy framework to sequentially select proper actions in different situations until all targets are picked out. Moreover, we introduce an active pushing mechanism based on a novel metric, namely Target-Centric Dispersion Degree (TCDD) for better grasping. TCDD describes whether the targets are isolated from the surrounding objects. With this metric, the robot becomes more focused on the actions around the targets and push irrelevant objects away. Experimental results on both simulated environment and real-world environment show that our proposed system outperforms several baseline approaches, which also has the capability to be generalized to new scenarios.

## 1. Introduction

“Picking out the impurities” is one of the typical scenarios in production line which is both time consuming and laborious for the workers. Typical applications of picking out the impurities includes separating slates from coal in coal production line, rejecting stones from mushroom in mushroom sorting, etc. In these situations, there are great needs for the robots to substitute humans to finish the work. Although robot grasping has been widely applied in production line, most of the existing systems only work in structured environments which is unable to handle dynamic situations, especially in picking out diverse objects in dense clutter [1].

Recently, target-oriented grasping and push-grasping have attracted great attention in robotics. References [2–4] mainly focused on picking up an object of a specific, user-selected semantic category, which is also called semantic grasping. To adapt quickly to target-oriented tasks, target information, such as accurate target masks [5] which is often obtained by pre-trained segmentation module with expensive labeling must be effectively combined. While in the problem of grasping the impurities, it is difficult to pre-define all the categories of the impurities to obtain the segmentation masks.

The challenge of grasping the impurities does not only lie in perception, but also with the manipulation capabilities. In early studies, the problem of manipulation capabilities was mainly addressed by hand-crafted features and pre-defining the sequences of actions [6, 7]. The hard-coded heuristics limit the types of synergistic behaviors between multiple primitive actions which can be performed. Recently, reinforcement learning methods are introduced into robot manipulation. Many studies were focused on



**Figure 1.** The robotic grasping impurities system. The system could automatically discover and grasp the impurities in dense clutter with the synergies between pushing and grasping actions.

training policies with multiple primitive actions through self-supervised trial and error [8, 9]. Although great progress has been achieved, there are still many challenges remaining to be solved when applying the synergies of different primitive actions to target-oriented grasping, for example, useless pushing actions in regions with no target and the low efficiency of grasping [10].

In this article, we propose a novel target-oriented push-grasping system which could actively discover and grasp the targets in dense clutter with the synergies between pushing and grasping actions as shown in Fig. 1. The main contributions are summarized as follows:

- We present a target-oriented attention module, which includes target saliency detection and density-based occluded-region inference. The attention module is not only able to locate the visible targets, but also predicts the regions where the targets are most likely to be occluded. Through the attention module, the robot can quickly discover where are the targets and perform policies with primitive actions to efficiently grasp the targets.
- We propose an active pushing mechanism based on a novel metric, namely Target-Centric Dispersion Degree (TCDD). TCDD is used to estimate the dispersion degree around the targets and to calculate the rewards for the pushing network. With the novel metric, the pushing network is trained to actively guide the robot to separate the target from other non-target objects and reduce the rate of useless pushes.
- We introduce an integrated push-grasping system framework for the robot to perform the synergies between pushing and grasping actions with Deep Q-Learning Network (DQN). The robot could properly select pushes or grasps to discover the impurities, separate them from others, and then grasp them out. Experimental results demonstrate the superior performance of our approach when picking out impurities in dense clutter.

## 2. Related Work

Grasping is one of the key research directions in the field of robotics which has been widely studied for decades [11–13]. Marwan et al. [13] thoroughly reviewed recent approaches for robot reaching and grasping. Early work of robotic grasping was mainly focused on analytical methods and 3D reasoning to predict the grasp locations and configurations [14–16]. Although analytical methods make grasping feasible, it requires physical modeling of the grasping objects which has high computational complexity and is difficult to apply to unstructured real-world environments. More recent data-driven methods explore the prospects of training model-agnostic deep grasping policies [17–20]. These methods detect grasps by exploiting learned visual features without explicitly using object-specific knowledge (e.g., shape, pose, dynamics) [21, 22]. Chu et al. [23] proposed a deep network architecture which predicts multiple grasp candidates in situations when none, single or multiple objects are in the view. The identification of grasp configurations for objects is broken down into a grasp detection process followed by a more refined grasp orientation classification process in ref. [23]. Fang et al. [24] contributed a large-scale grasp pose detection dataset namely GraspNet-1 Billion which contains 97, 280 RGB-D images with more than 1 billion grasp poses. Based on this dataset, they proposed an end-to-end grasp pose prediction network which achieved state-of-the-art performance in real-world experiments. Compared with analytical methods, data-driven methods learn grasping without considering the physical modeling of the objects. Our approach is fully self-supervised by trial and error.

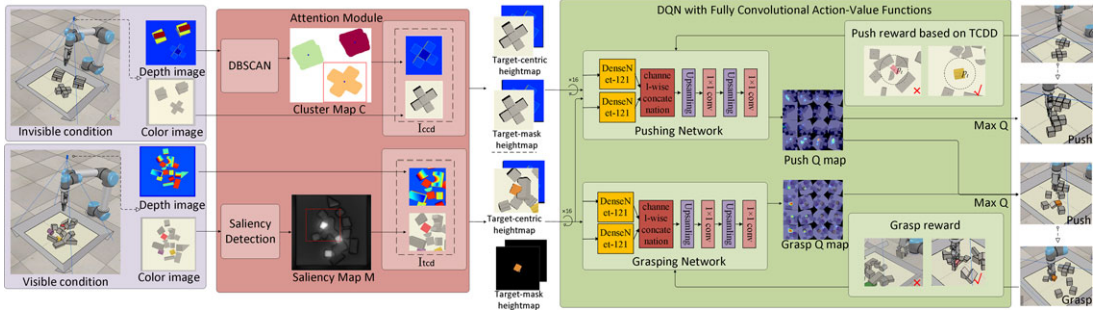
To effectively grasp objects in cluttered environments, many methods are focused on combining prehensile and non-prehensile manipulation policies. Most of the early work used hand-crafted features and pre-defined sequences of actions [6, 7, 25]. Recently, reinforcement learning is introduced to train end-to-end deep networks to learn complementary prehensile and non-prehensile manipulation policies [8, 9, 26]. Target-oriented grasping [27–29] cares more about the targets. It is often assumed that the targets are visible in target-oriented grasping since the visibility of the targets are required to learn object representations through interactions [27] or semantic segmentation [28]. Danielczuk et al. [4] proposed an action-heuristic method to select actions between grasping, sucking and pushing to retrieve targets occluded by clutter and further introduced X-Ray [5] which is able to efficiently extract the target object from a heap where it is fully occluded or partially occluded. The networks are trained with a dataset of 100 k RGB-D images labeled with occluded target objects. Yang et al. [30] separately learned a Bayesian-based policy to search for the targets and a classifier-based policy to coordinate the target-oriented pushing and grasping to grasp the targets in clutter.

Different from common target-oriented grasping where the categories of the grasping targets are specific, our work is focused on picking out the “impurities” in dense clutter. In our approach, the grasping targets are not limited to specific classes but are visually different from other objects in the clutter. This scenario exists more widely in industrial production line.

## 3. Problem Formulation

In target-oriented grasping, the goal is to grasp the target ( $X^*$ ) from physical environment ( $E$ ) containing various objects  $X$  in the task scope  $H$ . The push-grasping problem is then formulated as a Markov Decision Process (MDP) defined by tuples  $(S, A, T, R, \gamma)$ :

- State ( $s_t \in S$ ): In a bounded environment  $E$  containing  $N$  objects, the state  $s_t$  is represented as a RGB-D heightmap image of the environment. The RGB-D heightmap image shown in Fig. 1 is obtained by transforming the RGB-D image to be orthographically back-project upwards in the gravity direction [26]. In our work, the workspace of the robot is set as a 448 mm  $\times$  448 mm surface, which is evenly divided into 224  $\times$  224 action grids. The resolution of the heightmap in the workspace is 2 mm<sup>2</sup>.
- Action ( $a_t \in A$ ): A fixed set of parameterized motion primitives. We define two actions, push and grasp, in our system. The action  $a \in \{grasp, push\}$  is parametrized as a vector  $(x, y, z, \psi)$ , where



**Figure 2.** System pipeline. The system first obtains RGB-D images which are then fed into the attention module to discover the targets or predict the locations where the targets are likely to be occluded. The RGB-D images are then re-projected onto orthographic RGB-D heightmaps which are rotated with 16 different orientations. The rotated heightmaps are fed into two DenseNet tower structures and convolutional network to get the pixel-wise  $Q$  value tables. The two DenseNet networks correspond to the pushing and grasping actions, respectively. The system will select the grasping or pushing action with the highest  $Q$  value in the action-value table.

$(x, y, z)$  denotes the center position of the gripper.  $\psi \in [0, 2\pi]$  denotes the rotation of the gripper in the table plane. If the robot chooses to execute the grasp action, the gripper moves to  $(x, y, z)$  and rotates  $\psi$ , then closes the fingers. If the robot chooses to execute the push action, the gripper will first close the fingers, then moves to  $(x, y, z)$  and make a linear movement of 10 cm in length along the direction  $\psi$ .

- Transition ( $T$ ): Transition probability distribution in task environment,  $P: S \times S \times A \rightarrow [0, 1]$ . In model-free methods,  $T$  is unknown and the robot directly interacts with the environment.
- Reward ( $R$ ):  $r_t = r(s_t, a_t) \in R$  is defined as a binary reward to indicate whether the target is successfully grasped or the pushing actions are effective based on the metric TCDD.
- Future discount ( $\gamma$ ): The goal in reinforcement learning is to learn an optimal policy  $\pi^*$  to maximize the future reward  $R_t = \sum_{i=t}^T \gamma^{i-t} r_i$  at time  $t$  and with the discount factor  $\gamma \in [0, 1]$ . Our future discount  $\gamma$  is set as constant 0.5.

#### 4. Overview of the System

First as shown in Fig. 2, the RGB-D images are obtained from a fixed-mounted camera and are then fed into the attention module. In the attention module, the saliency detection submodule indicates the area where the target is included with high saliency. The density-based occluded-region inference submodule predicts the clutters with high densities are the places where the targets are likely to be occluded. If the target is in the view, the attention module will output a cropped image centered on the location with the highest saliency. If no target is found, the attention module will output a cropped image centered on the location with the highest density as the candidate area to discover the target.

Second, under the condition that no target is found, the pushing network with the output of the attention module as input would calculate the best pushing action to scatter objects in the designated local region. The robot repeats the action of pushing until the density of the objects in the view becomes uniform, which means the target objects are unlikely to be occluded. Under the condition that the targets are found, the pushing and grasping networks are both activated. Through the synergy between pushing and grasping, the target objects are isolated from surrounding objects until grasped. The above process is repeated until all targets in the environment are picked out.

**Algorithm 1. Attention Module.**


---

```

1: Input: RGB image  $I_c$ , depth image  $I_d$ 
2: Output: Target-centric RGB and depth image  $I_{tcd}$ , target mask  $I_m$  or cluster-centric RGB and depth
   image  $I_{ccd}$ , target mask  $I_{dm}$ 
3:  $(m_0, \dots, m_k) \subset M \leftarrow \text{VisualSaliencyDetection}(I_c)$ 
4:  $(c_0, \dots, c_k) \subset C \leftarrow \text{DBSCAN}(I_d)$ 
5: if  $M \neq \emptyset$  then
6:    $m_t = \max(m_0, \dots, m_k)$ 
7:    $I_{tcd} \leftarrow \text{CenterCrop}(m_t, I_c, I_d)$ 
8:    $I_m \leftarrow \text{Crop}(m_t)$ 
9: else
10:   $c_t = \max(c_0, \dots, c_k)$ 
11:   $I_{ccd} \leftarrow \text{CenterCrop}(c_t, I_c, I_d)$ 
12:   $I_{dm} \leftarrow \text{Crop}(c_t)$ 
13: end if

```

---

The pushing and grasping network uses fully convolutional action-value function (FCAVF) to map heightmap images to action-value tables in a Q-learning framework. The synergy between pushing and grasping actions is achieved by selecting the primitive action with the highest Q value from the pixel-wise Q table.

## 5. Method

In the following, we first introduce the details of the attention module in our proposed system. Then, we present our target-centric push–grasp synergy framework.

### 5.1. Attention module

Our attention module is designed with two components: target saliency detection and density-based occluded-region inference. These two components work in parallel in the proposed system. The saliency detection module has a higher priority over the occluded-region inference module. That is, the attention module will output the location of the clutter only if no target is found in current view. Algorithm 1 summarizes the details of the attention module.

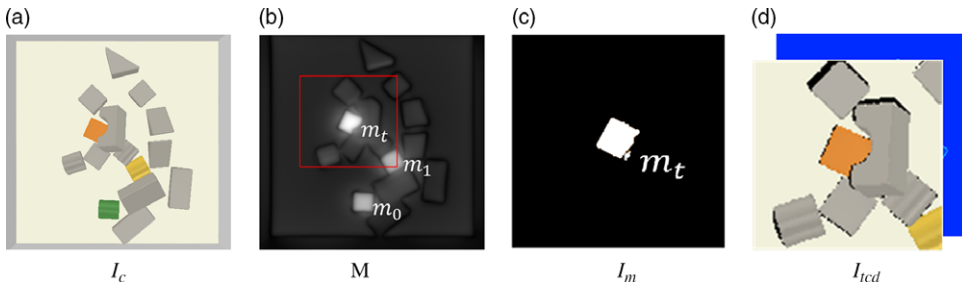
#### 5.1.1. Target saliency detection

Although the impurities are difficult to clean up if they are occluded by other objects, it is easy to identify when there is no occlusion. Since non-target objects are of the same type, the impurities are visually different in appearance with the surrounding objects. Based on this intuition, we introduce visual saliency detection to detect the salient objects in the view.

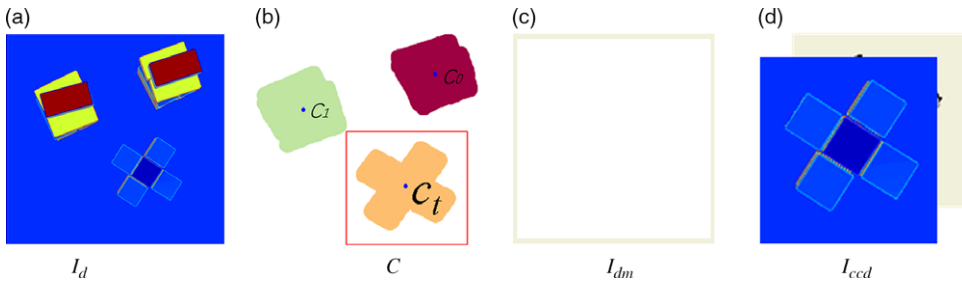
Visual saliency detection [31] simulates the mechanism of human visual attention in that salient objects could be easily identified in human visual system. Through visual saliency detection, we are able to focus on targets in highly salient regions which are apparently different from other objects in the view. In our approach, we introduce VOCUS2 [32] to obtain the saliency map. VOCUS2 follows FIT-based approach [33] which achieves the state-of-the-art performance for salient object segmentation. With VOCUS2, we obtain competitive results for salient object segmentation with the weighted f-measure. Moreover, the real-time performance can be guaranteed.

As shown in Fig. 3, we first get the saliency map  $M$  from the color image  $I_c$ . There may exist multiple salient regions  $(m_0, \dots, m_k)$ . The target mask image  $I_m$  centered on the region of the highest saliency  $m_t$





**Figure 3.** Example of target saliency detection. (a) The color image  $I_c$  in the view; (b) The saliency map  $M$ ; (c) The target mask image  $I_m$ ; (d) Target-centric color and depth image  $I_{ccd}$ .



**Figure 4.** Example of density-based occluded region inference. (a) The depth image  $I_d$  in the view. (b) The clustering results  $C$ . (c) The all-one mask image  $I_{dm}$ . (d) The cluster-centric color and depth image  $I_{ccd}$ .

is then returned.  $m_t$  is of a size of  $112 \times 112$  pixels. The RGB image  $I_c$  and the corresponding depth image are cropped into  $I_{ccd}$  centered on the target with the same size of  $I_m$ .  $I_m$  and  $I_{ccd}$  are used as the input to the pushing and grasping networks.

### 5.1.2. Density-based occluded region inference

If no target is visible in the cluttered scene, it is necessary to determine if the targets are occluded in some places. In dense clutter, the objects are scattered around in the scene which exhibits different density distributions of objects in different regions. Obviously, the higher density the region is, the higher possibility that the targets are occluded. To this end, we use DBSCAN [34] (Density-Based Spatial Clustering of Application with Noise) to obtain the clustering results of objects in the view based on the depth image.

Different from other clustering methods, DBSCAN defines a cluster as the largest set of densely connected points. DBSCAN is able to discover clusters of arbitrary shapes in spatial database with noise. As shown in Fig. 4, with DBSCAN, we can obtain the clusters  $C$  in the depth image  $I_d$ . The largest cluster  $C_t$  will be first selected as the region to explore. The all-one mask  $I_{dm}$  centered on the cluster  $c_t$  will be output.  $I_{dm}$  has the same size of  $I_m$ . The depth image  $I_d$  with the corresponding RGB image is then cropped into the image  $I_{ccd}$  centered on the cluster with the same size of  $I_{dm}$ .  $I_{dm}$  and  $I_{ccd}$  are also used as the input to the pushing and grasping networks. As shown in Fig. 5, if the height of the cluster in the scene is lower than a certain threshold or the area of the cluster is smaller than a certain threshold, the scene is considered to be in the state of no target presented.

## 5.2. Target-centric push–grasp synergy

### 5.2.1. DQN with fully convolutional action-value functions (FCAVF)

DQN [35] with Fully Convolutional Action-Value Function [26] uses a feed-forward fully convolutional network  $Q^\pi(s, a; \theta)$  as the Q function estimator to approximate the Q function in pixel. The weight of

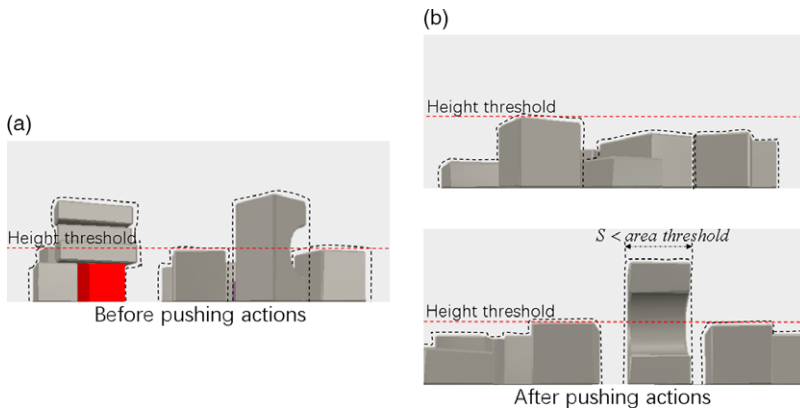


Figure 5. Density-based occluded region inference.

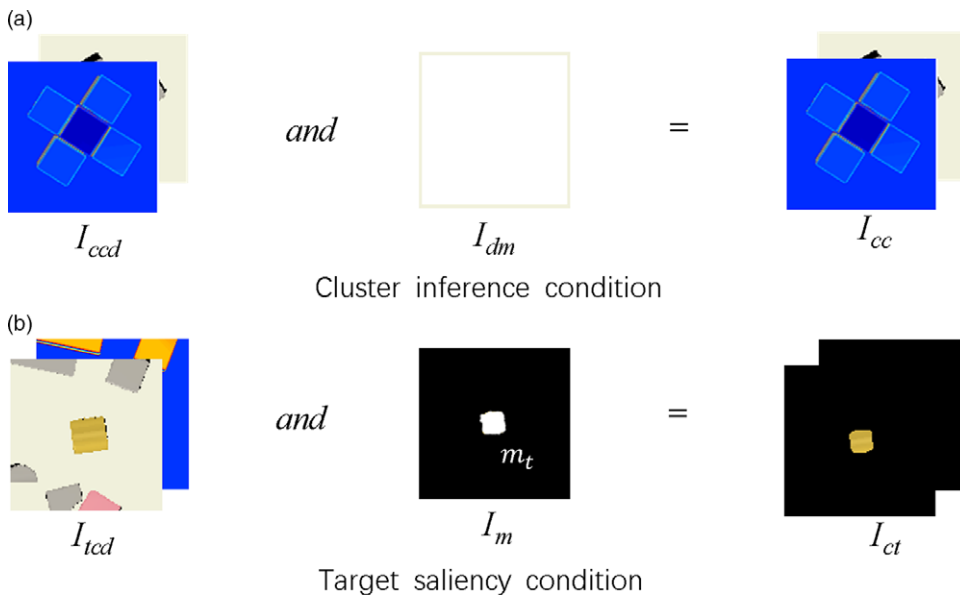


Figure 6. The generation of target color and depth images.

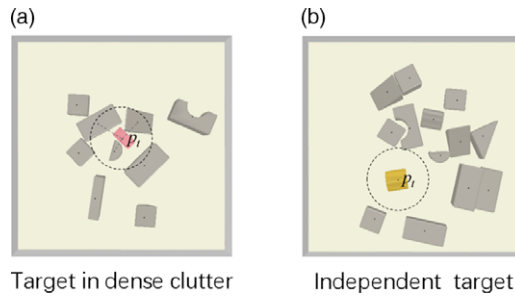
the network  $\theta$  is updated through gradient descent as

$$\theta_{t+1} \leftarrow \theta_t + \alpha(y_t^Q - Q(s_t, a_t; \theta_t)) \nabla_{\theta} Q(s_t, a_t; \theta) \tag{1}$$

where  $y_t^Q = r + \gamma \max_a Q(s', a'; \theta_{t-1})$  is the Q-target value calculated by the target network.

As shown in Fig. 6, in our approach, the target mask  $I_{dm}$  and  $I_m$  obtained from the attention module are first transformed into target color and depth images  $I_{cc}$  and  $I_{ct}$  by combining the cropped RGB and depth image  $I_{ccd}$  and  $I_{tcd}$  through the pixel-wise *and* operation. Then, with target-centric RGB and depth images  $I_{ccd}$  and  $I_{tcd}$ , we obtain two orthographic RGB-D heightmap images (target-mask heightmap and target-centric heightmap as shown in Fig. 2) after reprojection. In case of occluded-region inference, the target-mask heightmap is the same as the target-centric heightmap.

The pushing and grasping networks are trained to take two RGB-D heightmap images as input and infer pixel-wise predictions of Q values. The input heightmaps are first rotated into 16 orientations. Only horizontal pushes (to the right) and grasps are considered in the rotated images. The RGB channels of the heightmaps combined with the channel-wise cloned depth channel are two separate inputs into



**Figure 7.** Target-centric dispersion degree (TCDD). (a) shows a target in dense clutter and (b) shows an isolated target.

the two DenseNet towers [36], which are followed by a channel-wise concatenation and 2 additional convolutional layers with bilinear interpolation (as shown in Fig. 2). Finally, the total output is 32 pixel-wise maps of Q values (16 for pushes in different directions and 16 for grasps in different orientations). Q value at each pixel represents the expected return for performing the pushing or grasping action at the corresponding location and rotation angle. The action which has the highest Q value in the pixel-wise maps will be chosen.

5.2.2. Learning active pushing for grasping

For the pushing action in target-agnostic grasping task, only considering the changes of the environment after pushing as the positive reward might be an effective solution [26]. However, it cannot guarantee to isolate the targets from the densely surroundings which sometimes even makes the objects closer to each other. To this end, we introduce a novel metric of Target-Centric Dispersion Degree (TCDD), which is specifically designed to calculate the spread-out degree of target from the surrounding objects. With this metric, the pushing network could be trained to guide the robot to separate the target from the surrounding objects efficiently.

During training in simulation, we first calculate the distance between the target and the surrounding objects in the scene. If the distance is greater than the finger-opening distance of the gripper  $\eta$ , it indicates that the target will not affect other objects when being grasped. Therefore, the dispersion distance between objects is defined as follows (Fig. 7(a)):

$$d(p_t, p_j) = \begin{cases} \|p_t - p_j\|_2, & \text{if } \|p_t - p_j\|_2 < \eta \\ \eta, & \text{if } \|p_t - p_j\|_2 \geq \eta \end{cases} \tag{2}$$

Here  $p_t$  and  $p_j$  denote the center coordinates of the target  $t$  and object  $j$ , respectively. If there are  $K$  objects in the scene, the TCDD is defined as

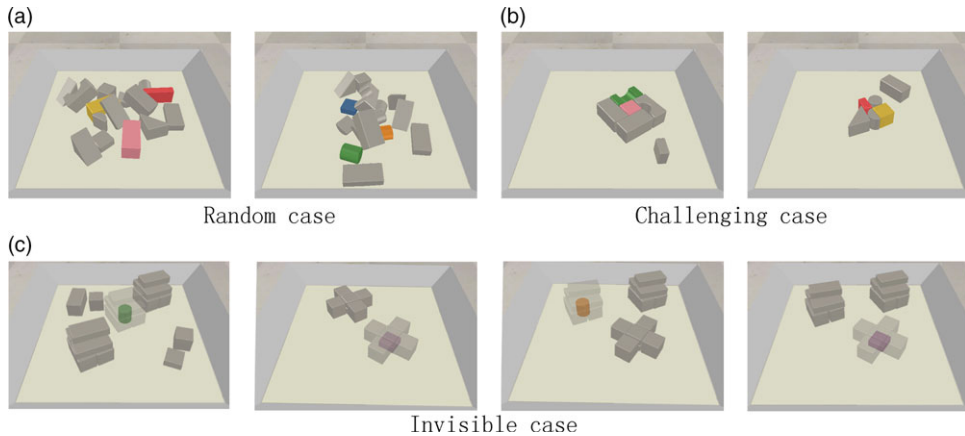
$$\alpha_t = \sum_{j=1}^{K-1} \frac{d(p_t, p_j)}{(K-1)\eta} \tag{3}$$

According to the formula defined above, the larger the  $\alpha_t$  is, the greater the dispersion degree of the target object  $t$  in the scene. If  $\alpha_t = 1$ , the target  $t$  is considered to be *isolated* (Fig. 7(b)) which can be grasped freely. Based on TCDD, we train the pushing network in simulation and apply the pushing policy in testing.

5.2.3. Reward

The rewards for grasping and pushing actions are specified separately. We set the reward for grasping  $R_g(s_t, s_{t+1}) = 1$  if the grasp is successful (the target is grasped at a specified location). The reward for pushing  $R_p(s_t, s_{t+1}) = 0.5$  if  $\alpha_{t+1} - \alpha_t > \delta$ , which means the objects become more scattered after the





**Figure 8.** Three experimental settings in simulated environment.

pushing action. For other conditions, we consider the actions are failed and the rewards are set to 0. To reduce the effect of noise, we set  $\delta = 0.005$ .

## 6. Experimental Results and Analysis

Extensive experiments are carried out to evaluate the performance of our proposed system. The goals of our experiments are to answer the following questions: (1) Can our proposed system solve the problem of grasping impurities in different experimental settings? (2) Does our approach improve task performance compared to baseline methods? (3) How are the different components of our proposed system influence the performance in different settings?

### 6.1. Simulation experimental setup

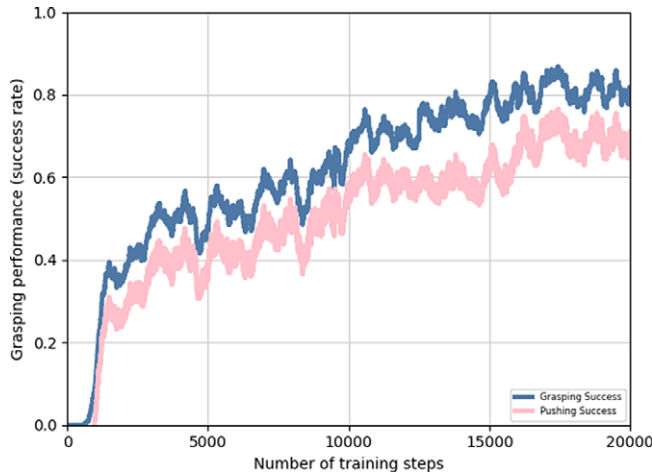
We use a UR5 robot arm with a parallel gripper RG2 and a statically mounted camera in V-REP [36] with bullet V2.83 in simulation. The RG2 gripper is a flexible collaborative gripper with 2 kg payload and built-in Quick Changer up to 110 mm stroke.  $n$  targets and  $m$  basic blocks are added in a workspace to simulate the dense cluttered environment. The target objects are of different colors from other blocks to simulate the impurities. The robot needs to grasp the target objects via a sequence of pushing and grasping actions. The scene is reset and objects are randomly dropped if all targets are grasped.

To validate the effectiveness of our approach, we set up three experimental scenarios: (1) Random Case. As shown in Fig. 8(a), objects are randomly dropped onto the table. (2) Challenging Case. Objects are laid closely side by side, which is difficult to grasp if there is no synergy of actions. An example of the configuration is shown in Fig. 8(b). (3) Invisible Case. There are multiple closely arranged heaps, which makes the targets invisible. This case is shown in Fig. 8(c).

For each test, we execute 20 test runs for each scene and use two metrics to evaluate the performance: (1) the average success rate of grasp per-completion, (2) the action efficiency, which is defined as  $N_{object}/N_{action}$  to describe the efficiency of the policy to complete the task.  $N_{object}$  is the number of targets in testing and  $N_{action}$  is the number of actions per-completion.

### 6.2. Training details

In our experiments, the models are trained by self-supervision. The agent takes the output of the attention module as input to learn the policies. The policies of pushing and grasping are learned jointly through



**Figure 9.** Performance of push–grasp synergies policy. The blue line indicates the grasping success rates and the pink line indicates the pushing success rates.

trial and error. Our push–grasp action policy is trained to minimize the temporal difference error  $\delta_t = Q(s_t, a_t; \theta_t) - y_t^{\theta_t^-}$  at each iteration  $i$  using the Huber loss function:

$$L_i = \begin{cases} \frac{1}{2} \delta_t^2, & \text{if } \delta_t < 1 \\ \|\delta_t\|, & \text{otherwise} \end{cases} \quad (4)$$

where  $\theta_t$  is the parameter of the neural network at iteration  $t$ . The target network parameter  $\theta_t^-$  is fixed between individual updates. At iteration  $t$ , the gradient is pass only at the single pixel  $p$  where the executed action  $a_t$  was executed.

We train our model in PyTorch with version 0.3. Our model is optimized by Stochastic Gradient Descent (SGD) with the momentum of 0.9 and a weight decay of  $2 \times 10^{-5}$ . The learning rate is set to  $10^{-5}$ . The network uses a prioritized experience replay [37] with stochastic rank-based prioritization in training. We train our model only in the “Random Case” and test in all three settings. Figure 9 illustrates the training process of our policy learning. The performance in training is measured by the success rate of the last 300 attempts.

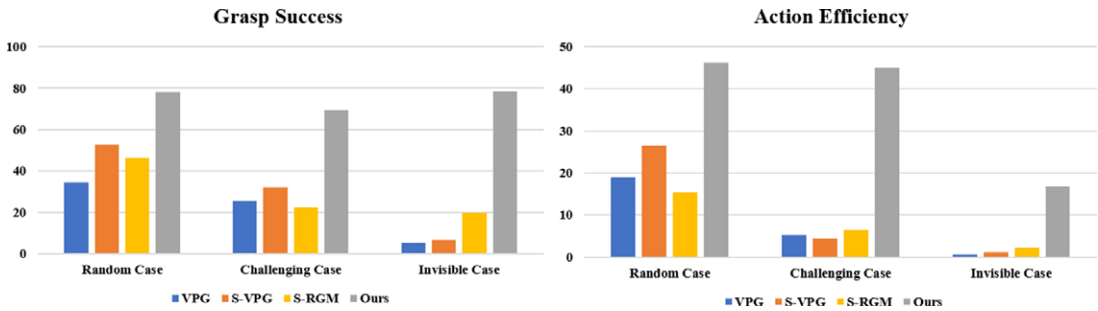
### 6.3. Simulation results

We verify the effectiveness of our proposed push-grasping system by evaluating the performance in a series of tests. In these test runs, the system needs to successfully pick up the targets from the desktop in different scene settings. For better training, we change the scene if the same action fails for 3 consecutive times.

#### 6.3.1. Performance

We compare the performance of our approach with three baselines: (1) VPG [26] uses push–grasp synergies to grasp objects. (2) S-VPG is an extension of VPG, which uses the visual saliency detection mask as the input to both pushing and grasping networks. The action with the highest Q value is executed. (3) S-RGM, which adopts BASNet [38] for visual saliency detection and predicts the 5-dimensional grasp rectangle of the objects proposed in ref. [23]. A heuristic pushing mechanism is used if the grasping action is failed.

The comparison results in three experimental settings are summerized in Fig. 10. In random and challenging cases, VPG performs poorly and shows success rates of 34.6% and 25.4%. Since VPG is



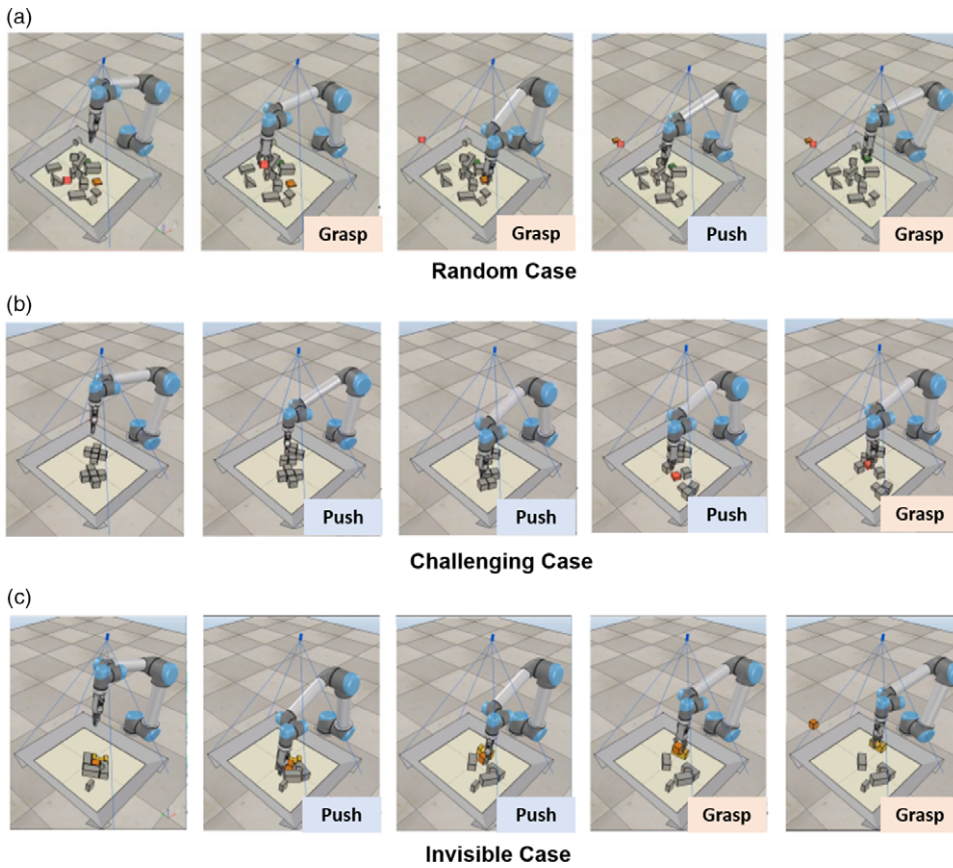
**Figure 10.** Comparison with baseline methods. The left figure shows the success rate of grasp and the right figure shows the action efficiency.

target agnostic, the targets are picked out by chance. S-VPG has better performance compared with VPG and achieves success rates of 52.6% and 32% in random case and challenging case, respectively. The reason lies in that S-VPG has the target information. S-VPG directly applies the target mask on the results of VPG, which cannot be applied in case of unknown targets. However, S-VPG lacks the ability to actively isolate the target object from surrounding objects, which makes it inefficient in our task. S-RGM achieves similar performance as S-VPG in random case (46.2%) and VPG in challenging case (22.4%). The saliency detection in S-RGM could guide the robot to focus on the targets. However, the predicted grasp rectangle is not accurate enough for successful grasping which lowers the success rate. Our proposed method achieves the best performance by more than 25% and 37% in terms of task success rate compared with S-VPG in random and challenging cases. The action efficiency of all three baselines are not very high (18.9%, 26.5% and 15.4% in random case and 5.3%, 4.4% and 6.5% in challenging case). The action efficiency becomes lower as the difficulty level increases from random case to challenging case. While ours remain high efficiency (46.3% in random case and 45.1% in challenging case) and does not influenced by the change of the scenario.

In invisible case, the performance of VPG and S-VPG drops significantly where the success rates are about 5% and the action efficiencies are only about 1%. This is mainly due to the fact that these two methods do not have effective mechanism to find the targets actively under the condition that no target is available. S-RGM achieves better performance (20% success rate) compared with VPG and S-VPG since S-RGM adopts a similar heuristic mechanism as our proposed method to guide the robot to push the area with higher cluster density, which increases the possibility of finding the targets. However, S-RGM is still not effective enough (2.2% action efficiency). Our method has the highest success rate (78.5%) and achieves almost similar or better performance as the random case and the challenging case. The action efficiency drops to (16.8%) because of more pushing actions are needed for finding the targets. In a nutshell, our method focuses on discovering and grasping the targets where the number of useless actions in irrelevant regions is reduced. With our approach, the robot is able to fully utilize active pushing actions to improve the grasping success rate of the targets.

### 6.3.2. Ablation studies

We perform ablation experiments to evaluate the importance of each component in our proposed method. The following conditions are considered: (1) No Target Saliency Detection (No-TSD). Without the saliency detection, the whole RGB and depth images of the scene with  $224 \times 224$  pixels are re-projected onto orthographic RGB-D heightmaps and serve as the representation of the current state. (2) No Occluded Region Inference (No-OI). If there is no target detected in the scene, images centered on the maximum value of the saliency map are replaced as the input to the push-grasping policy. (3) No TCDD (No-T). The metric for training pushing network is used to evaluate whether the scene changes before and after the pushing action [26]. If the difference after the action execution exceeds a certain threshold  $\tau$  ( $\tau \geq 300$  pixels), the push reward is set to 1, otherwise it is 0.



**Figure 11.** Policy execution in simulation (a) Random case; (b) Challenging case; (c) Invisible case.

The detailed experimental results are shown in Table I. In the case of NO-TSD, the robot performs poorly in all cases. With the increasing difficulty levels of the cases, the grasp success rate quickly drops from 34.6% to 5.1%. The action efficiency drops from 18.9% to 0.6%. This is due to the fact that the policy from No-TSD lacks the target information. The problem decays to be target-agnostic. In the case of No-OI, the grasp success rate (53.6% and 49.7%) and action efficiency (34.5% and 29.3%) are not greatly influenced in random and challenging cases, but drop greatly in invisible case (15.7% and 8.9%). The reason is that No-OI lacks the ability to predict the candidate regions where the targets are likely to be occluded. In the case of No-T, the policy performance is better than that of No-OI which remains satisfactory performance in random and challenging cases. However, the performance is still not satisfactory in the invisible case. This is due to the ineffectiveness of the metric if the targets are occluded by the surrounding objects. Therefore, Target Saliency Detection (TSD) is the most important component in our method. TSD could guide the robot to quickly locate the targets. Occluded region inference also greatly helps the robot to discover targets in invisible case. TCDD shows its effectiveness when grasping the targets from the heaps. Figure 11 shows the simulation results in all three cases.

**6.4. Results on a physical robot**

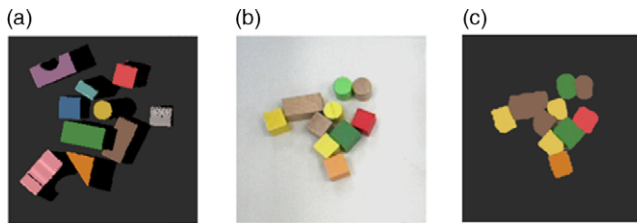
We also validate the effectiveness of the learned policy on a real-world UR5 robotic arm with a parallel gripper Robotiq 85. The maximum grip stroke of Robotiq85 is 85 mm (adjustable) and has a similar

**Table I.** Results of ablation studies.

Method	Grasp success			Action efficiency		
	Random case	Challenging case	Invisible case	Random case	Challenging case	Invisible case
No-TSD	34.6%	25.4%	5.1%	18.9%	5.3%	0.6%
No-OI	53.6%	49.7%	15.7%	34.5%	29.3%	8.9%
No-T	64.5%	62.2%	27.2%	35.8%	36.3%	7.5%
Ours	78.1%	69.4%	78.5%	46.3%	45.1%	16.8%

**Table II.** Experimental results in real-world environment.

Method	Toy blocks		Real-world objects	
	Grasp Success	Action Efficiency	Grasp Success	Action Efficiency
Random Case	79.8%	47.9%	75.7%	45.6%
Challenging Case	64.9%	38.3%	59.3%	33.3%
Invisible Case	71.6%	23.0%	67.7%	18.9%

**Figure 12.** Results of domain adaptation. (a) Images in simulation; (b) Images from real-world environment; (c) Images adapted after domain adaptation.

structure with RG 2, which makes it possible to directly apply the grasping policy learned from simulation to real-world experiments. To better transfer the learned policy from simulation environment to the real-world environment, we adopt the domain adaptation method [39] to deal with the reality gap. As shown in Fig. 12, we can see that the image from real-world environment (b) becomes more similar in color distribution as the image in simulation (a) after domain adaptation. With domain adaption, our learned policy trained in simulation can be transferred into the real-world experiments directly.

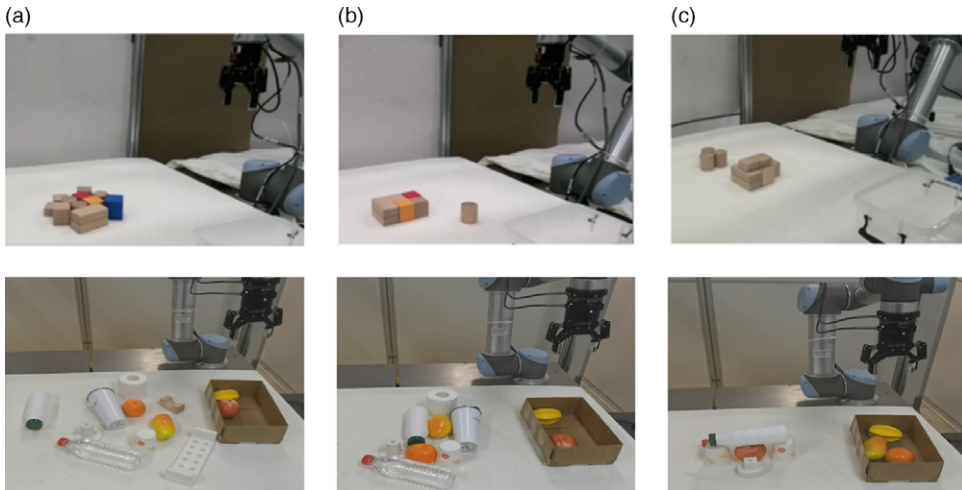
The learned policy are evaluated in all three cases in two real-world experiments (toy blocks and real-world objects) shown in Fig. 13. Through changing the initial locations and distributions of objects, we test 10 groups of experiments in each case and the maximum step is set to 15 in each group. Experimental results are shown in Table II.

From Table II, we can see that our method performs stable and satisfactory in both two real-world experiments. In random case, our method uses less pushes than other cases. In challenging case, the robot needs more pushes for better grasping. In invisible case, the robot needs further more pushes to scatter the surrounding objects to discover the target. Figures 14 and 15 show the results of the learned policy in three settings.

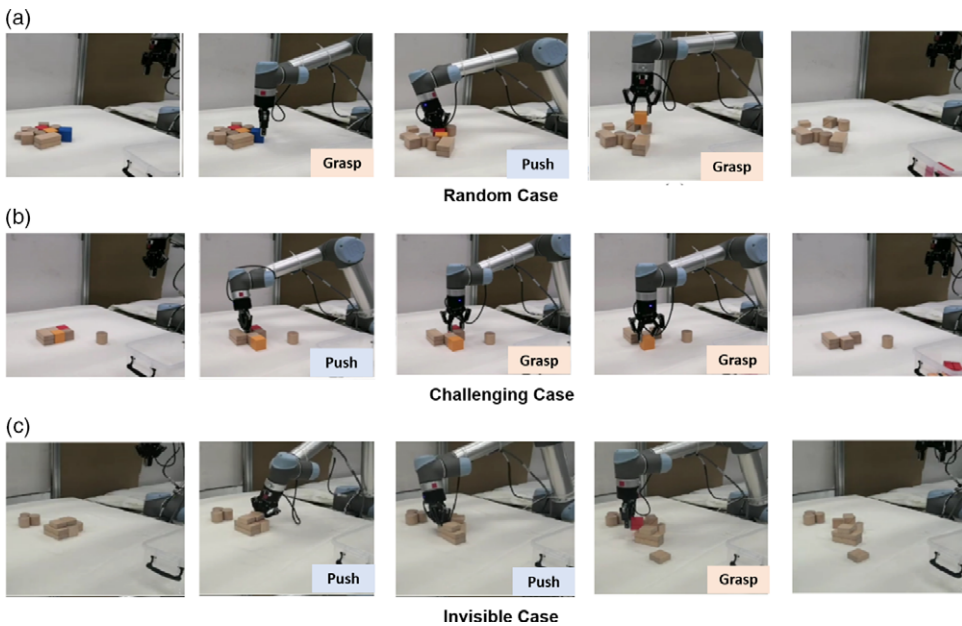
## 7. Conclusions

In this article, we presented the problem of “Picking out the impurities” in dense clutter where an attention-based push-grasp synergy system with deep reinforcement learning is proposed. In our





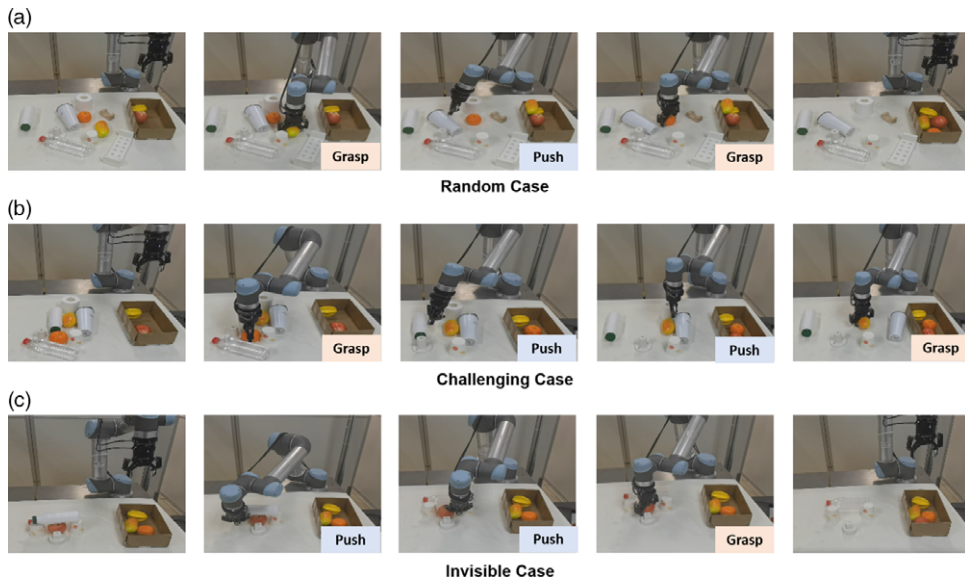
**Figure 13.** Experimental settings in real-world experiments (a) Random case; (b) Challenging case; (c) Invisible case. The first row shows experiment setting of toy blocks. The second row shows experiment setting of real-world objects.



**Figure 14.** Policy execution in toy block experiment (a) Random case; (b) Challenging case; (c) Invisible case.

approach, the proposed attention model is able to quickly locate the targets and also predicts the most likely regions where the targets are occluded. The new metric TCDD effectively guides the robot to isolate the target from the surroundings. The number of useless pushes is also reduced. Our system is trained in simulation with self-supervision. Extensive experimental results in simulation and real-world environment demonstrate that our proposed method can effectively pick out the impurities in dense clutter with better exploration efficiency and higher grasp success rate than baseline approaches. Moreover, the trained policies can be generalized to challenging environments with noises.





**Figure 15.** Policy execution in real-world object experiment (a) Random case; (b) Challenging case; (c) Invisible case.

**Acknowledgement.** This work was supported by the National Key R&D Program of China (grant 2019YFB1311901) and the National Natural Science Foundation of China under Grant U1713222 and 61773378.

**Conflicts of Interest.** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## References

- [1] Y. Deng, X. Guo and Y. Wei, et al., “Deep Reinforcement Learning for Robotic Pushing and Picking in Cluttered Environment,” *In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019) pp. 619–626.
- [2] E. Jang, S. Vijayanarasimhan, P. Pastor, J. Ibarz and S. Levine, “End-to-End Learning of Semantic Grasping,” *In: Proceedings of The 1st Conference on Robot Learning, CoRL 2017*, (vol. **78**, PMLR, 2017) pp. 119–132.
- [3] M. Kiatos and Malassiotis, S., “Robust Object Grasping in Clutter via Singulation,” *In: International Conference on Robotics and Automation (ICRA)* (2019) pp. 1596–1600.
- [4] M. Danielczuk, A. Kurenkov, A. Balakrishna, et al., “Mechanical Search: Multi-Step Retrieval of a Target Object Occluded by Clutter,” *In: International Conference on Robotics and Automation (ICRA)* (2019) pp. 1614–1621.
- [5] M. Danielczuk, A. Angelova, V. Vanhoucke and K. Goldberg, “X-Ray: Mechanical Search for an Occluded Object by Minimizing Support of Learned Occupancy Distributions,” *In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2020) pp. 9577–9584.
- [6] M. Gupta and G. S. Sukhatme, “Using Manipulation Primitives for Brick Sorting in Clutter,” *In: IEEE International Conference on Robotics and Automation (ICRA)* (2012) pp. 3883–3889.
- [7] M. Dogar and S. Srinivasa, “A planning framework for non-prehensile manipulation under clutter and uncertainty,” *Auton. Robot.* **33**(3), 217–236 (2012).
- [8] A. Boularias, J. A. Bagnell and A. Stentz, “Learning to Manipulate Unknown Objects in Clutter by Reinforcement,” *In: The Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI 15)* (AAAI Press, 2015) pp. 1336–1342.
- [9] J. Ibarz, D. Kalashnikov, P. Pastor, et al., “Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation,” *In: Proceedings of The 2nd Conference on Robot Learning, CoRL 2018* (vol. **87**, PMLR, 2018) pp. 651–673.
- [10] H. Liu, Y. Yuan, Y. Deng, et al., “Active Affordance Exploration for Robot Grasping,” *In: International Conference on Intelligent Robotics and Applications (ICIRA)* (2019) pp. 426–438.
- [11] A. Sahbani, S. El-Khoury and P. Bidaud, “An overview of 3D object grasp synthesis algorithms,” *Robot. Auton. Syst.* **60**(3), 326–336 (2012).
- [12] J. Bohg, A. Morales, T. Asfour and D. Kragic, “Data-driven grasp synthesis—A survey,” *IEEE Trans. Robot.* **30**(2), 289–309 (2014).
- [13] Q. Marwan, S. Chua and L. Kwek, “Comprehensive review on reaching and grasping of objects in robotics,” *Robotica* **39**(10), 1849–1882 (2021).

- [14] D. Ding and Y. Liu, Wang, S., “Computing, “3-D Optimal Form-Closure Grasps,” **In: International Conference on Robotics and Automation (ICRA) (2000)** pp. 3573–3578.
- [15] C. Mitash, A. Boularias and K. Bekris, “Robust 6D Object Pose Estimation with Stochastic Congruent Sets,” **In: British Machine Vision Conference (2018)**.
- [16] J. Ponce, S. Sullivan, J. D. Boissonnat and J.-P. Merlet, “On Characterizing and Computing Three- and Four-Finger Force-Closure Grasps of Polyhedral Objects,” **In: IEEE International Conference on Robotics and Automation (ICRA) (1993)** pp. 821–827.
- [17] Y. Jiang, S. Moseson and A. Saxena, “Efficient Grasping from RGBD Images: Learning Using a New Rectangle Representation,” **In: IEEE International Conference on Robotics and Automation (ICRA) (2011)** pp. 3304–3311.
- [18] A. Zeng, K. Yu, S. Song, et al., “Multi-view Self-Supervised Deep Learning for 6D Pose Estimation in the Amazon Picking Challenge,” **In: IEEE International Conference on Robotics and Automation (ICRA) (2017)** pp. 1386–1383.
- [19] L. Pinto and A. Gupta, “Supersizing Self-supervision: Learning to Grasp from 50K tries and 700 Robot Hours,” **In: IEEE International Conference on Robotics and Automation (ICRA) (2016)** pp. 3406–3413.
- [20] A. Zeng, S. Song, K. Yu, et al., “Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching,” *Int. J. Robot. Res.* **56**(1), 1–16 (2019).
- [21] C. Choi, W. Schwarting, J. DelPreto and A. Rus, “Learning object grasping for soft robot hands,” *IEEE Robot. Automat. Lett.* **3**(3), 2370–2377 (2018).
- [22] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy and K. Goldberg, “Dex-Net 3.0: Computing Robust Vacuum Suction Grasp Targets in Point Clouds Using a New Analytic Model and Deep Learning,” **In: IEEE International Conference on Robotics and Automation (ICRA) (2018)** pp. 5620–5627.
- [23] F. Chu, R. Xu and A. V. Patricio, “Real-World multi-object, multi-grasp detection,” *IEEE Robot. Automat. Lett.* **3**(4), 3355–3362 (2018).
- [24] H. Fang, C. Wang, M. Gou and C. Lu, “GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping,” **In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)** pp. 11444–11453.
- [25] D. Omrčen, C. Böge, T. Asfour, A. Ude and R. Dillmann, “Autonomous Acquisition of Pushing Actions to Support Object Grasping with a Humanoid Robot,” **In: IEEE-RAS International Conference on Humanoid Robots (2009)** pp. 277–283.
- [26] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez and T. Funkhouser, “Learning Synergies Between Pushing and Grasping with Self-Supervised Deep Reinforcement Learning,” **In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2018)** pp. 4238–4245.
- [27] E. Jang, C. Devin, V. Vanhoucke and S. Levine, “Grasp2Vec: Learning Object Representations from Self-Supervised Grasping,” **In: Proceedings of The 2nd Conference on Robot Learning, CoRL 2018 (vol. 87, PMLR, 2018)** pp. 99–112.
- [28] K. Fang, Y. Bai, S. Hinterstoisser, S. Savarese and M. Kalakrishnan, “Multi-Task Domain Adaptation for Deep Learning of Instance Grasping from Simulation,” **In: IEEE International Conference on Robotics and Automation (ICRA) (2018)** pp. 3516–3523.
- [29] A. Kurenkov, J. Taglic, R. Kulkarni, et al., “Visuomotor Mechanical Search: Learning to Retrieve Target Objects in Clutter,” **In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2020)** pp. 8408–8414.
- [30] Y. Yang, H. Liang and C. Choi, “A deep learning approach to grasping the invisible,” *IEEE Robot. Automat. Lett.* **5**(2), 2232–2239 (2020).
- [31] R. Cong, J. Lei, H. Fu, M. Cheng, W. Lin and Q. Huang, “Review of visual saliency detection with comprehensive information,” *IEEE Trans. Circ. Syst. Vid.* **29**(10), 2941–2959 (2019).
- [32] S. Frintrop, T. Werner and G. M. García, “Traditional Saliency Reloaded: A Good Old Model in New Shape,” **In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)** pp. 82–90.
- [33] L. Itti, C. Koch and E. Niebur, “A model of saliency-based visual attention for rapid scene analysis,” *IEEE Trans. Patt. Anal.* **20**(11), 1254–1259 (1998).
- [34] H. Bäcklund, A. Hedblom and N. Neijman, A Density-Based Spatial Clustering of Application with Noise,” **In: Linköpings Universitet–ITN, Data Mining TNM033 (2011)** pp. 11–30.
- [35] V. Mnih, K. Kavukcuoglu, D. Silver et al., “Human-level control through deep reinforcement learning,” *Nature* **518**, 529–533 (2015).
- [36] G. Huang, Z. Liu, L. V. D. Maaten and K. Q. Weinberger, “Densely Connected Convolutional Networks,” **In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)** pp. 2261–2269.
- [37] E. Rohmer, S. P. N. Singh and M. Freese, “V-REP: A Versatile and Scalable Robot Simulation Framework,” **In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)** pp. 1321–1326.
- [38] X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan and M. Jagersand, “BASNet: Boundary-Aware Salient Object Detection,” **In: IEEE Conference on Computer Vision and Patter Recognition (CVPR) (2019)** pp. 7479–7489.
- [39] T. Schaul, J. Quan, I. Antonoglou and D. Silver, “Prioritized Experience Replay,” **In: International Conference on Learning Representations (ICLR) (2016)**.
- [40] K. Bousmalis, N. Silberman, D. Dohan, et al. “Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks,” **In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)** pp. 3722–3731.